**问题描述**

|  |  |
| --- | --- |
| 试题编号： | 2015-12-2 |
| 试题名称： | 消除类游戏 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　消除类游戏是深受大众欢迎的一种游戏，游戏在一个包含有*n*行*m*列的游戏棋盘上进行，棋盘的每一行每一列的方格上放着一个有颜色的棋子，当一行或一列上有连续三个或更多的相同颜色的棋子时，这些棋子都被消除。当有多处可以被消除时，这些地方的棋子将同时被消除。 　　现在给你一个*n*行*m*列的棋盘，棋盘中的每一个方格上有一个棋子，请给出经过一次消除后的棋盘。 　　请注意：一个棋子可能在某一行和某一列同时被消除。  **输入格式**  　　输入的第一行包含两个整数*n*, *m*，用空格分隔，分别表示棋盘的行数和列数。 　　接下来*n*行，每行*m*个整数，用空格分隔，分别表示每一个方格中的棋子的颜色。颜色使用1至9编号。  **输出格式**  　　输出*n*行，每行*m*个整数，相邻的整数之间使用一个空格分隔，表示经过一次消除后的棋盘。如果一个方格中的棋子被消除，则对应的方格输出0，否则输出棋子的颜色编号。  **样例输入**  4 5 2 2 3 1 2 3 4 5 1 4 2 3 2 1 3 2 2 2 4 4  **样例输出**  2 2 3 0 2 3 4 5 0 4 2 3 2 0 3 0 0 0 4 4  **样例说明**  　　棋盘中第4列的1和第4行的2可以被消除，其他的方格中的棋子均保留。  **样例输入**  4 5 2 2 3 1 2 3 1 1 1 1 2 3 2 1 3 2 2 3 3 3  **样例输出**  2 2 3 0 2 3 0 0 0 0 2 3 2 0 3 2 2 0 0 0  **样例说明**  　　棋盘中所有的1以及最后一行的3可以被同时消除，其他的方格中的棋子均保留。  **评测用例规模与约定**  　　所有的评测用例满足：1 ≤ *n*, *m* ≤ 30。 |
| 代码程序： | #include<stdio.h>  int main()  {  int n,m;  int M[35][35];  int H[35][35];  int L[35][35];  scanf("%d %d",&n,&m);  for(int i = 0; i<n;i++)  {  for(int j = 0; j<m;j++)  {  scanf("%d",&M[i][j]);  H[i][j] = M[i][j];  L[i][j] = M[i][j];  }  }  for(int i = 0; i<n;i++)  {  int count = 1;  int now = H[i][0];  int j = 1;  while(j<m)  {  while(j<m&&H[i][j] == now)  {  j++;  count++;  }  if(count>=3)  {  for(int k =j-count ;k<=j-1;k++)  {  H[i][k] = 0;  }  }  now = H[i][j];  count = 0;  }  }  for(int i = 0;i<m;i++)  {  int count = 1;  int now = L[0][i];  int j = 1;  while(j<n)  {  while(j<n&&L[j][i] == now)  {  j++;  count++;  }  if(count>=3)  {  for(int k =j-count ;k<=j-1;k++)  {  L[k][i] = 0;  }  }  now = L[j][i];  count = 0;  }  }  for(int i = 0; i<n;i++)  {  for(int j = 0;j<m;j++)  {  if(H[i][j] == 0 || L[i][j] == 0)  {  M[i][j] = 0;  }  }  }  for(int i = 0; i<n;i++)  {  for(int j = 0;j<m;j++)  {  printf("%d",M[i][j]);  if(j<m-1)  {  printf(" ");  }  }  printf("\n");  }  return 0;  } |